AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently amended) A method for mapping ~~data in a markup language document to an object model~~ between a markup language document and an object model, the method comprising the steps of;

receiving a mapping request for mapping ~~data in a markup language document having data architecture into an object model~~ between a markup language document and an object model, wherein the mapping request includes a key for identifying the markup language document, wherein the mapping between the markup language document and the object model includes:; ~~and~~

mapping data in the markup language document having data architecture into an object model; and

mapping one or more objects in an object model into a markup language document;

mapping, in response to the mapping request, either the data directly into the object model, or the objects directly into the markup language document, using mapping meta-data which defines how the data architecture of the markup language document maps to the object model, wherein the mapping step obtains the markup language document using the key;

wherein mapping the data in the markup language document directly into the object model using the mapping meta-data enables the method to support inheritance or relationships in the object model.

2 (Canceled).

3. (Original) The method as claimed in claim 1, wherein the markup language document has one or more elements containing data, the object model has one or more object classes, each object class has one or more attributes that correspond to the elements, and the step of mapping includes a step of populating the attributes with the data of the corresponding elements based on the mapping meta-data.

4. (Original) The method as claimed in claim 1, wherein the markup language document has one or more elements containing data, the object model has one or more object classes, each object class has one or more attributes that correspond to the elements and the step of mapping includes;

a step of generating a row structure corresponding to the markup language elements of the markup language document;

a step of converting the row structure into one or more objects corresponding to the elements; and

a step of populating attributes of the converted objects with the data of the elements based on the mapping meta-data.

5. (Original) The method as claimed in claim 3, wherein the markup language document further has at least one element containing one or more other elements and the mapping step inserts, based on the mapping meta-data, a value representing the relation between the at least one element and the one or more other elements into an attribute of the object model to represent a relationship between objects corresponding to the at least one element and the one or more other elements.

6. (Original) The method as claimed in claim 5, wherein the at least one element contains a single element containing data and the mapping step inserts a value representing the relation between the at least one element and the single element into an attribute of the object model that represents a one-to-one relationship between objects that correspond to the at least one element and the single element.

7. (Original) The method as claimed in claim 5, wherein the at least one element contains a single element containing a pointer to another element in a different markup language document and the mapping step inserts a value representing the relation between the at least one element and the single element into an attribute of the object model that represents an aggregate one-to-one relationship between objects that correspond to the at least one element and the single element.

8. (Original) The method as claimed in claim 5, wherein the at least one element contains multiple elements containing data and the mapping step inserts values representing the relation between the at least one element and the multiple elements into attributes of the object model that represent one-to-many relationships between objects that correspond to the at least one element and the multiple elements.

9. (Original) The method as claimed in claim 5, wherein the at least one element contains multiple elements containing pointers to elements in one or more different markup language documents and the mapping step inserts values representing the relation between the at least one element and the multiple elements into attributes of the object model that represent aggregate one-to-many

relationships between objects that correspond to the at least one element and the multiple elements.

10. (Original) The method as claimed in claim 1 further comprising a step of obtaining the mapping meta-data prior to the mapping step.

11. (Previously presented) The method as claimed in claim 10, wherein the obtaining step is carried out during initialization of a system for executing the receiving step and the mapping step.

12. (Original) The method as claimed in claim 1, wherein the markup language document has one or more elements, the object model has one or more object classes, and the mapping meta-data includes mapping information regarding one of the elements and the corresponding object class.

13. (Original) The method as claimed in claim 1, wherein the markup language document has one or more elements, the object model has one or more object classes, each object class has one or more attributes, the mapping meta-data includes mapping information regarding one of the elements that contains data and the corresponding attribute, and the mapping step maps the data of the one of the elements into the corresponding attributes based on the mapping information.

14. (Original) The method as claimed in claim 1, wherein the markup language document is a document in which each element is defined by indicators.

15. (Original) The method as claimed in claim 14, wherein the markup language document is an eXtensible Markup Language (XML) document.

16 (Canceled).

17 (Canceled).

18. (Currently amended) The method as claimed in ~~claim 16~~ claim 1, wherein the object model has one or more object classes containing the one or more objects to be mapped, each object class has one or more attributes, and the step of the mapping includes a step of creating one or more elements of the markup language document corresponding to the one or more objects by inserting values of the attributes into the elements based on the mapping meta-data.

19. (Currently amended) The method as claimed in ~~claim 16~~ claim 1, wherein the object model has one or more object classes containing one or more objects to be mapped, each object class has one or more attributes, and the step of mapping includes:

a step of generating a row structure corresponding to the one or more objects based on the mapping meta-data; and

a step of creating elements of the markup language document based on the row structure.

20. (Original) The method as claimed in claim 18, wherein the attributes include an attribute representing a relationship between the objects and the mapping step maps a value representing the relationship between the elements.

21. (Original) The method as claimed in claim 20, wherein an attribute represents a one-to-one relationship between a source object and a target object and the mapping step maps a value representing the one-to-one relationship to an

element containing another element for containing data that corresponds to data of the target object.

22. (Original) The method as claimed in claim 20, wherein an attribute represents an aggregate one-to-one relationship between a source object and a target object and the mapping step maps a value representing the aggregate one-to-one relationship to an element containing another element for containing a pointer to point to another element in a different markup language document that contains data corresponding to data of the target object.

23. (Original) The method as claimed in claim 20, wherein an attribute represents a one-to-many relationship between a source object and multiple target objects and the mapping step maps values representing the one-to-many relationship to an element containing multiple other elements for containing data that correspond to data of the multiple target objects.

24. (Original) The method as claimed in claim 20, wherein an attribute represents an aggregate one-to-many relationship between a source object and multiple target objects and the mapping step maps values representing the aggregate one-to-many relationship to an element containing multiple other elements for containing pointers to points other elements in one or more different markup language documents that contain data corresponding to data of the target objects.

25. (Currently amended) The method as claimed in ~~claim 16~~ claim 1 further comprising a step of obtaining the mapping meta-data prior to the mapping step.

26. (Previously presented) The method as claimed in claim 25, wherein the obtaining step is carried out during initialization of a system for executing the receiving step and the mapping step.

27. (Currently amended) The method as claimed in ~~claim 16~~ claim 1, wherein the markup language document has one or more elements, the object model has one or more object classes, and the mapping meta-data includes information regarding the object class and the corresponding one of the elements.

28. (Currently amended) The method as claimed in ~~claim 16~~ claim 1, wherein the markup language document has one or more elements, the object model has one or more object classes, the object class has one or more attributes, and the mapping meta-data includes information regarding one of the attributes and the corresponding one of the elements.

29. (Currently amended) The method as claimed in ~~claim 16~~ claim 1, wherein the markup language document is a document in which each element is defined by indicators.

30. (Original) The method as claimed in claim 29, wherein the markup language document is a XML document.

31. (Previously presented) A mapping manager for mapping between a markup language document and an object model, the mapping manager comprising:

an executor for receiving a mapping request for mapping between a markup language document having data architecture and an object model, wherein

the mapping request includes a key for identifying the markup language document; and

a mapping executor for directly mapping, in response to the mapping request, between data of the markup language document and objects of the object model using mapping meta-data which defines how the data architecture of the markup language document maps to the object model, wherein the mapping step obtains the markup language document using the key;

wherein mapping the data in the markup language document directly into the object model using the mapping meta-data enables the mapping executor to support inheritance or relationships in the object model.

32 (Canceled).

33. (Original) The manager as claimed in claim 31, wherein the markup language document has one or more elements, the object model has one or more object classes, each object class has one or more attributes, and the mapping executor includes a mapping unit for populating the attributes with the data of the elements based on the mapping meta-data.

34. (Original) The manager as claimed in claim 31, wherein the markup language document has one or more elements, the object model has one or more object classes, each object class has one or more attributes, and the mapping executor includes:

a generator for generating a row structure corresponding to the markup language elements;

a converter for converting one or more objects based on the row structure; and

a mapping unit for populating attributes of the converted objects with the data of the elements based on the mapping meta-data.

35. (Original) The manager as claimed in claim 33, wherein the markup language document further has at least one element containing one or more other elements, and the mapping unit inserts, based on the mapping meta-data, a value representing the relation between the at least one element and the one or more other elements into an attribute of the object model to represent a relationship between objects corresponding to the at least one element and the one or more other elements.

36. (Original) The manager as claimed in claim 35, wherein the at least one element contains a single element containing data and the mapping unit inserts a value representing a relation between the at least one element and the single element into an attribute of the object model that represents a one-to-one relationship between objects that corresponds to the at least one element and the single element.

37. (Original) The manager as claimed in claim 35, wherein the at least one element contains a single element containing a pointer to another element in a different markup language document, and the mapping unit inserts a value representing the relation between the at least one element and the single element into an attribute of the object model that represents an aggregate one-to-one relationship between objects that corresponds to the at least one element and the single element.

38. (Original) The manager as claimed in claim 35, wherein the at least one element contains multiple elements containing data, and the mapping unit inserts values representing the relation between the at least one element and the multiple elements into attributes of the object model that represent one-to-many relationships between objects that corresponds to the at least one element and the multiple elements.

39. (Original) The manager as claimed in claim 35, wherein the at least one element contains multiple elements containing pointers to elements in one or more different markup language documents, and the mapping unit inserts values representing the relation between the at least one element and the multiple elements into attributes of the object model that represent aggregate one-to-many relationships between objects that correspond to the at least one element and the multiple elements.

40. (Original) The manager as claimed in claim 31, wherein the mapping executor includes a mapping unit for obtaining the mapping meta-data.

41. (Original) The manager as claimed in claim 31, wherein the markup language document has one or more elements, the object model has one or more object classes, and the mapping executor includes a mapping unit for handling a mapping between one of the elements and the corresponding object class.

42. (Original) The manager as claimed in claim 31, wherein the markup language document has one or more elements, the object model has one or more object classes, each object class has one or more attributes, and the mapping

executor includes a mapping unit for handling a mapping between one of the elements and the corresponding attribute.

43. (Original) The manager as claimed in claim 31, wherein the markup language document has one or more elements, the object model has one or more object classes, each object class has one or more attributes, the attributes include an attribute representing a relationship between the objects, the mapping executor includes a relationship mapping unit for handling a mapping of a relationship between the objects, and the relationship represents a relation between the elements.

44. (Original) The manager as claimed in claim 31, wherein the object model has one or more object classes, each object class has one or more attributes, and the mapping executor includes a mapping unit for creating one or more elements corresponding to the attributes by inserting values of the attributes based on the mapping meta-data.

45. (Original) The manager as claimed in claim 31, wherein the markup language document is a document in which each element is defined by indicators.

46. (Original) The manager as claimed in claim 45, wherein the markup language document is an XML document.

47. (Previously presented) A mapping system for mapping between a markup language document and an object model, the mapping system comprising:

an executor for receiving a mapping request for mapping between a markup language document having data architecture and an object model, wherein

the mapping request includes a key for identifying the markup language document;

a storage for storing mapping meta-data which defines how the data architecture of the markup language document maps to the object model; and

a mapping executor for directly mapping, in response to the mapping request, between data of the markup language document and an object of the object model using the mapping meta-data, wherein the mapping step obtains the markup language document using the key;

wherein mapping the data in the markup language document directly into the object model using the mapping meta-data enables the mapping executor to support inheritance or relationships in the object model.

48. (Original) The system as claimed in claim 47 wherein the mapping executor includes a mapping unit for obtaining the mapping meta-data from the storage.

49. (Original) The system as claimed in claim 47, wherein the mapping storage obtains the mapping meta-data prior to an operation of the mapping executor.

50. (Original) The system as claimed in claim 47, wherein the mapping storage obtains the mapping meta-data during initialization of the system.

51. (Original) The system as claimed in claim 47 further comprising a runtime interface to accept the mapping request from an application.

52. (Original) The system as claimed in claim 47, wherein the markup language document has one or more elements, the object model has one or more object classes, the object class has one or more attributes, and the mapping executor includes a mapping unit for populating the attributes with the data associated with the elements based on the mapping meta-data.

53. (Original) The system as claimed in claim 47, wherein the object model has one or more object classes, each object class has one or more attributes, and the mapping executor includes a mapping unit for creating one or more elements corresponding to the attributes by inserting values of the attributes based on the mapping meta-data.

54. (Original) The system as claimed in claim 47, wherein the markup language document is a document in which each element is defined by indicators.

55. (Original) The system as claimed in claim 54, wherein the markup language document is an XML document.

56. (Currently amended) Computer readable media storing the instructions or statements for use in the execution in a computer of a method for mapping ~~data in a markup language document to an object model~~ between a markup language document and an object model, the method comprising the steps of;

receiving a mapping request for mapping ~~data in a markup language document having data architecture into an object model~~ between a markup language document and an object model, wherein the mapping request includes a key for identifying the markup language document, wherein the mapping between the markup language document and the object model includes:; ~~and~~

mapping data in the markup language document having data architecture into an object model; and

mapping one or more objects in an object model into a markup language document;

mapping, in response to the mapping request, either the data directly into the object model, or the objects directly into the markup language document, using mapping meta-data which defines how the data architecture of the markup language document maps to the object model, wherein the mapping step obtains the markup language document using the key;

wherein mapping the data in the markup language document directly into the object model using the mapping meta-data enables the method to support inheritance or relationships in the object model.

57 (Canceled).

58. (Currently amended) A computer program product for use in the execution in a computer of a method for mapping ~~data of a markup language document to an object model~~ between a markup language document and an object model, the computer program product comprising:

a module for receiving a mapping request for mapping ~~data in a markup language document having data architecture into an object model~~ between a markup language document and an object model, wherein the mapping request includes a key for identifying the markup language document, wherein the mapping between the markup language document and the object model includes:

mapping data in the markup language document having data architecture into an object model; and

mapping one or more objects in an object model into a markup language document; ~~and~~

a module for mapping, in response to the mapping request, either the data directly into the object model, or the objects directly into the markup language document, using mapping meta-data which defines how the data architecture of the markup language document maps to the object model, wherein the mapping step obtains the markup language document using the key;

wherein mapping the data in the markup language document directly into the object model using the mapping meta-data enables the module for mapping to support inheritance or relationships in the object model.

59 (Canceled).

60 (Canceled).

61 (Canceled).